ES6 top-up

ES6/2015

- Destructuring.
- The spread operator.
- Default arguments

Destructuring

- **Assigning the properties of an array or object to variables using syntax that looks similar to array or object literals.**

Instead of:

```
let nums = [10, 11, 12]
let v1 = nums[0]
let v2 = nums[1]
let v3 = nums[2]
```

Use:

```
let [v1, v2, v3] = nums
```

Instead of:

```
let obj = { alpha:100,
            beta: 'enterprise'}
let alpha = obj.alpha
let beta = obj.beta
```

Use :

```
let {alpha, beta} = obj;
```

- **See 01_*_destructuring.js**

The Spread operator (…)

- **Allows an** iterable **to expand in places where 0+ arguments are expected.**
  - **Iterable – Arrays, Objects.**
- **. . .ArrayRef; . . .objectRef**
- **See 02_*_spread.js**

Default arguments

```
function add(x = 1, y = 2) {
    return x + y;
}
console.log(add(5));                    // 7
console.log(add(undefined, 1));   // 2
console.log(add());                     // 3
```

JavaScript Object Notation

JSON

JSON

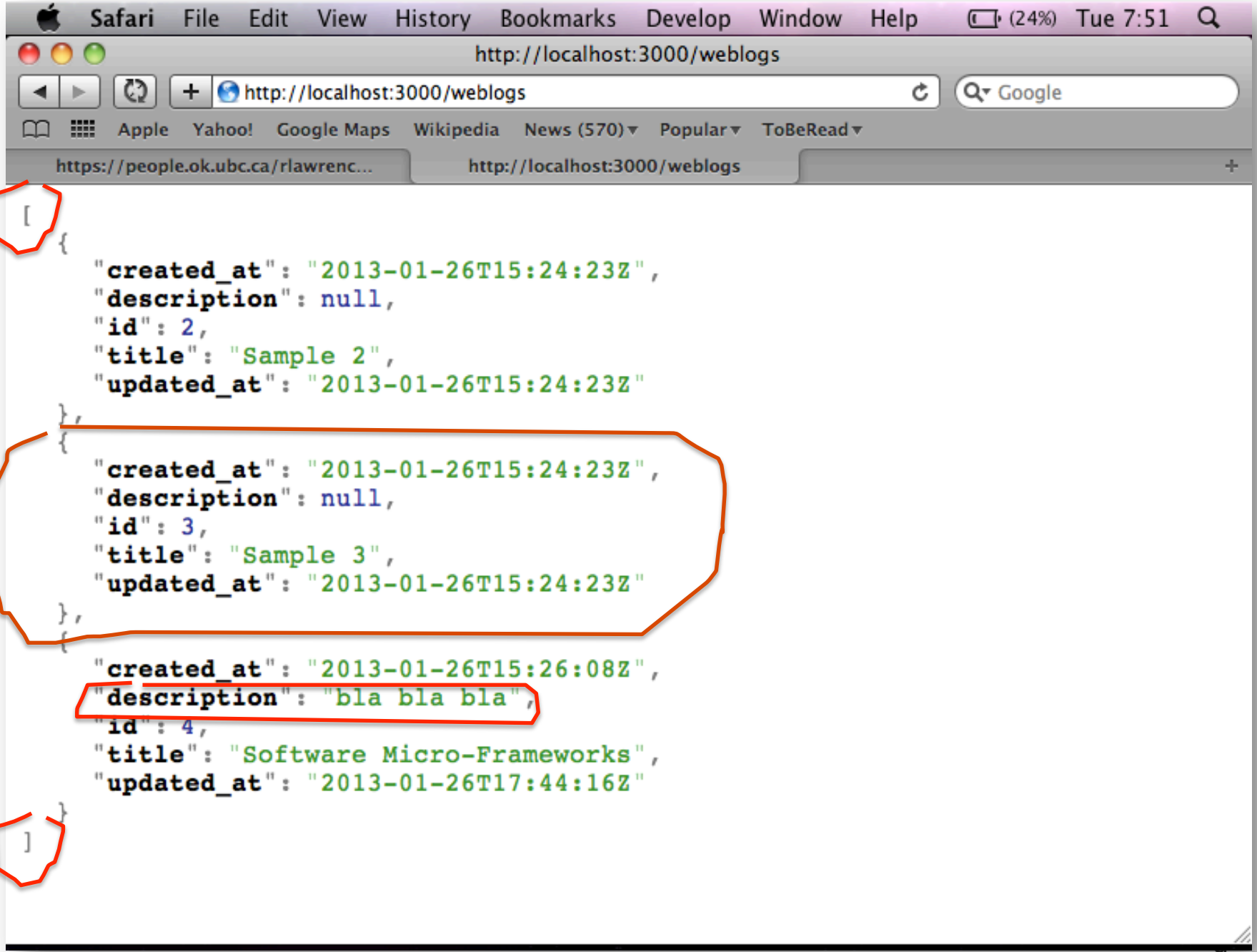
- **A standard for serializing data into text form.**
  - **Alternative to XML serialization.**

- **Advantages:**
  1. **Human-readable (like XML, but easier).**
  2. **Useful for data interchange between applications (like XML, but less verbose).**
  3. **Useful for representing and storing semi-structured data.**
     - **Unlike the Relational data model, which is only suited for structured data.**

- **JSON is no longer tied to JavaScript – lots of languages have JSON parsers.**

JSON

- **JSON constructs:**
  1. **Base Values:**
     - **number, strings (double quoted), boolean (true / false), null.**
  2. **Composite values:**
     a. **Objects: enclosed in { } and consist of set of key-value pairs.**
        – **Keys must be double-quoted strings ****
     b. **Arrays: enclosed in [ ] and are lists of values.**
     - **Objects and arrays can be nested.**


Safari File Edit View History Bookmarks Develop Window Help (24%) Tue 7:51
http://localhost:3000/weblogs
http://localhost:3000/weblogs
Google
Apple Yahoo! Google Maps Wikipedia News (570) Popular ToBeRead
https://people.ok.ubc.ca/rlawrenc...
http://localhost:3000/weblogs
[
{
"created_at": "2013-01-26T15:24:23Z",
"description": null,
"id": 2,
"title": "Sample 2",
"updated_at": "2013-01-26T15:24:23Z"
},
{
"created_at": "2013-01-26T15:24:23Z",
"description": null,
"id": 3,
"title": "Sample 3",
"updated_at": "2013-01-26T15:24:23Z"
},
{
"created_at": "2013-01-26T15:26:08Z",
"description": "bla bla bla",
"id": 4,
"title": "Software Micro-Frameworks",
"updated_at": "2013-01-26T17:44:16Z"
}
]

A Property.

```
{ "Books":
  [
    { "ISBN":"ISBN-0-13-713526-2",
      "Price":85,
      "Edition":3,
      "Title":"A First Course in Database Systems",
      "Authors":[ {"First_Name":"Jeffrey", "Last_Name":"Ullman"},
                  {"First_Name":"Jennifer", "Last_Name":"Widom"} ] }
    ,
    { "ISBN":"ISBN-0-13-815504-6",
      "Price":100,
      "Remark":"Buy this book bundled with 'A First Course' - a great deal!",
      "Title":"Database Systems:The Complete Book",
      "Authors":[ {"First_Name":"Hector", "Last_Name":"Garcia-Molina"},
                  {"First_Name":"Jeffrey", "Last_Name":"Ullman"},
                  {"First_Name":"Jennifer", "Last_Name":"Widom"} ] }
  ],
  "Magazines":
  [
    { "Title":"National Geographic",
      "Month":"January",
      "Year":2009 }
    ,
    { "Title":"Newsweek",
      "Month":"February",
      "Year":2009 }
  ]
}
```

Semi-structured

Relational model Vs JSON model

	JSON	Relational
Structure	Nested objects + arrays	Tables
Schema	Variable (and not required)	Fixed
Queries	Limited	SQL, RA
Ordering	Arrays are sorted	No
Systems	Used with programming languages and some NoSQL systems	Many commercial and open source systems

XML Vs JSON.

XML versus JSON

	XML	JSON
Verbosity	More	Less
Complexity	More	Less
Validity	DTDs widely used XSDs	JSON Schema not widely used
Prog. Interface	Clunky "Impedence mismatch"	More direct
Querying	XPath - XQuery XSLT -	JSON Path JSON Query

Stateless Functional components

- **Many components only require the** render **method.**
- **The lifecycle methods are redundant but still effect performance (inherited).**
- **Use** stateless functional components **(sfc) where possible.**

```
const ComponentName = (props) =>
    { …. body of render method …..}
```

- Legacy code - jscodeshift **tool transforms conventional (class-based) components to sfc.**

  $ npm install –g jscodeshift

  ….. Must also install transformer(s) seperately

  $ jscodeshift -t transforms/pure-component.js --useArrows=true --destructuring=true <path to source file>